

(19)日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号
特開2000-29709
(P2000-29709A)

(43)公開日 平成12年1月28日(2000.1.28)

(51)Int.Cl. ⁷	識別記号	F I	テマコード(参考)
G 0 6 F 9/44	5 5 2	G 0 6 F 9/44	5 5 2
15/16	6 2 0	15/16	6 2 0 W

審査請求 有 請求項の数26 O L (全 14 頁)

(21)出願番号 特願平11-153502
(22)出願日 平成11年6月1日(1999.6.1)
(31)優先権主張番号 09/089961
(32)優先日 平成10年6月3日(1998.6.3)
(33)優先権主張国 米国 (US)

(71)出願人 390009531
インターナショナル・ビジネス・マシー
ズ・コーポレーション
INTERNATIONAL BUSIN
ESS MACHINES CORPO
RATION
アメリカ合衆国10504、ニューヨーク州
アーモンク (番地なし)
(72)発明者 マイケル・マクナリー
アメリカ合衆国78730、テキサス州オース
ティン、モスヘッド・コープ 10503
(74)代理人 100086243
弁理士 坂口 博 (外1名)

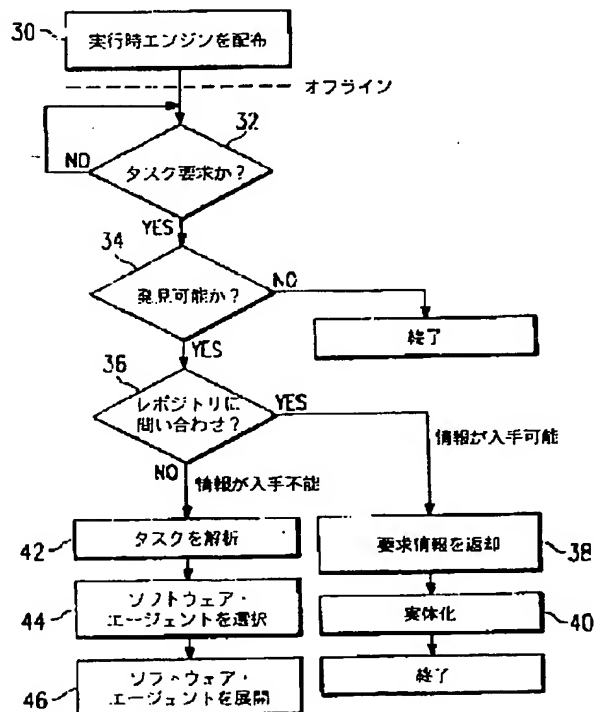
最終頁に続く

(54)【発明の名称】 分散コンピュータ環境内における発見のためのシステム、方法及びコンピュータ・プログラム製
品

(57)【要約】

【課題】 非常に多数のマシンが接続され、管理される分散コンピュータ企業環境において、発見(ディスカバリ)操作を実行すること。

【解決手段】 大規模分散企業が、複数のエンドポイント・マシンをサービスする管理サーバを含むコンピュータ資源を含む。管理基盤が、所与のエンドポイント・マシン上に展開される実行時エンジンを含む。管理サーバにおけるタスク展開要求に応答して、発見エージェントがコンピュータ・ネットワーク内に発動される。ソフトウェア・エージェントが実行時エンジンをサポートする所与のマシンに到来すると、エージェントが実行され、エンドポイントが特定のタスク展開の候補であるか否かを判断する。



BEST AVAILABLE COPY

【特許請求の範囲】

【請求項 1】複数のマシンにサービスする管理サーバを有する分散コンピュータ・ネットワークにおける発見（ディスカバリ）のための方法であって、実行時エンジンのインスタンスを前記マシンのサブセットに渡って展開し、前記分散コンピュータ・ネットワーク内で分散実行時環境を生成するステップと、所与のオペレーションにตอบสนองして、発見エージェントをソースから前記コンピュータ・ネットワーク内に展開するステップと、前記実行時エンジンのインスタンスをサポートする所与のマシンにおいて、前記実行時エンジンを用いて前記発見エージェントを実行し、発見操作を実行するステップとを含む、方法。

【請求項 2】前記発見操作が、タスク展開のために好適なマシンの識別、マシンに関連付けられる資源のセットの識別、マシン・タイプの識別、またはマシンにおける資源の所与の特性の識別を行う、請求項 1 記載の方法。

【請求項 3】マシンにおいて前記発見エージェントをクローン化するステップを含む、請求項 1 記載の方法。

【請求項 4】クローン化された発見エージェントを、前記コンピュータ・ネットワーク内の新たな経路に沿って発動するステップを含む、請求項 3 記載の方法。

【請求項 5】前記実行時環境が実行時エンジンを含み、前記発見エージェントが前記実行時エンジンにより実行可能な 1 つ以上のタスクのセットである、請求項 1 記載の方法。

【請求項 6】前記コンピュータ・ネットワークがインターネットであり、前記実行時エンジンがブラウザに関連付けられ、前記ソフトウェア・エージェントがアプレットである、請求項 5 記載の方法。

【請求項 7】複数のマシンにサービスする管理サーバを有する分散コンピュータ・ネットワークにおける発見（ディスカバリ）のための方法であって、実行時エンジンのインスタンスを前記マシンのサブセットに渡って展開し、前記分散コンピュータ・ネットワーク内で分散実行時環境を生成するステップと、タスク展開要求にตอบสนองして、発見エージェントを前記コンピュータ・ネットワーク内の所与のノード・セットに渡って移動するステップと、前記発見エージェントが受信される各ノードにおいて、前記発見エージェントを前記実行時環境で実行するステップであって、
a) 当該ノードのマシンが配布要求のターゲットであるか否かを判断するステップと、
b) 前記ノードに関連付けられ、前記配布要求の候補であるノードのサブセットを識別するステップと、
c) 前記発見エージェントを前記サブセットに展開するステップと、
d) 前記ターゲットが識別されるまで、または全てのネ

ットワーク経路が探索されるまで、前記ステップ a) 乃至 c) を繰り返すステップとを含む、前記発見エージェントを実行するステップとを含む、方法。

【請求項 8】前記配布要求を受信するマシンのリストを編成するステップを含む、請求項 7 記載の方法。

【請求項 9】複数のマシンにサービスする管理サーバを有する大規模分散コンピュータ・ネットワークにおけるタスク発見方法であって、前記マシンの所与のサブセットが以前に展開された実行時環境を含むものにおいて、
10 タスク展開要求にตอบสนองして、前記管理サーバから発見エージェントを前記コンピュータ・ネットワーク内に発動するステップと、

前記サブセット内の所与のマシンにおいて、該所与のマシン上の前記実行時環境を使用し、前記発見エージェントの 1 つを実行するステップと、

前記サブセットにおいてタスク展開を受信するマシンを識別する情報を、前記管理サーバに返却するステップとを含む、方法。

【請求項 10】前記発見エージェントの少なくとも 1 つが、前記タスク展開要求にもとづきカスタマイズされる、請求項 9 記載の方法。

【請求項 11】前記実行時環境が実行時エンジンを含み、各前記発見エージェントが、前記実行時エンジンにより実行可能な 1 つ以上のタスクのセットである、請求項 9 記載の方法。

【請求項 12】前記コンピュータ・ネットワークがインターネットであり、前記実行時エンジンがブラウザに関連付けられ、前記発見エージェントがアプレットである、請求項 11 記載の方法。

【請求項 13】管理基盤がサポートされる大規模分散コンピュータ・ネットワーク内で作用するタスク発見方法であって、

ソフトウェア・エージェントのセットから、少なくとも 1 つのソフトウェア・エージェントを選択するステップと、

タスク展開のためのターゲット・マシンを識別するために、前記コンピュータ・ネットワーク内の所与のノード・セットに渡り、選択したソフトウェア・エージェントを移動するステップとを含む、方法。

【請求項 14】前記選択するステップが、タスクの所与の特性を識別するステップと、前記所与の特性にもとづき、前記ソフトウェア・エージェントを選択するステップとを含む、請求項 13 記載の方法。

【請求項 15】前記移動するステップが、所与のノードにおいて、前記ソフトウェア・エージェントをクローン化するステップと、クローン化されたソフトウェア・エージェントをネットワーク経路に沿って移動するステップとを含む、請求項 13 記載の方法。

【請求項 16】前記管理基盤が、前記コンピュータ・ネットワークの所与のノードにおいて展開される実行時環境を含み、前記ソフトウェア・エージェントが、前記実行時環境により実行される 1 つ以上のタスクのセットである、請求項 13 記載の方法。

【請求項 17】前記コンピュータ・ネットワークがインターネットであり、前記実行時環境がブラウザに関連付けられ、前記ソフトウェア・エージェントがアプレットである、請求項 16 記載の方法。

【請求項 18】複数のエンドポイント・マシンにサービスする管理サーバを有する大規模分散企業内に接続され、発見操作を実行する装置であって、各々が所与のエンドポイント・マシンにおいてサポートされる、実行時エンジンの複数のインスタンスと、所与のオカレンスに回答して、所与のエンドポイント・マシンの実行時エンジンにより実行可能なソフトウェア・エージェントを選択する手段と、選択されたソフトウェア・エージェントを前記コンピュータ・ネットワーク内に展開し、発見操作を実行する手段とを含む、装置。

【請求項 19】複数のエンドポイント・マシンにサービスする管理サーバを有する大規模分散企業内に接続され、タスクを展開する発見システムであって、各々が所与のエンドポイント・マシンにおいてサポートされる、実行時エンジンの複数のインスタンスと、発見要求に回答して、所与のエンドポイント・マシンの実行時エンジンにより実行され、所与の基準に適合するマシンを識別する 1 つ以上のソフトウェア・エージェントのセットを、前記分散企業内にディスパッチする手段とを含む、システム。

【請求項 20】前記所与の基準が、エンドポイント・マシンが発見操作の候補であるか否かの判断である、請求項 19 記載のシステム。

【請求項 21】前記ソフトウェア・エージェントを生成する手段を含む、請求項 19 記載のシステム。

【請求項 22】前記生成する手段が、所与のソフトウェア・エージェントを前記発見操作の機能としてカスタマイズする手段を含む、請求項 21 記載のシステム。

【請求項 23】管理サーバを有する分散ネットワーク内に設置され、プロセッサ及びメモリを有するコンピュータで使用されるコンピュータ読出し可能媒体内のコンピュータ・プログラム製品であって、第 1 の操作の間に前記コンピュータにダウンロードされる実行時エンジンと、発見操作の間に前記コンピュータに展開され、前記実行時エンジンにより実行されて、前記コンピュータが所与の基準に適合するか否かを発見するソフトウェア・エージェントとを含む、コンピュータ・プログラム製品。

【請求項 24】前記ソフトウェア・エージェントが、前記発見操作の機能として選択される 1 つ以上のタスクを

含む、請求項 23 記載のコンピュータ・プログラム製品。

【請求項 25】前記実行時エンジンが前記コンピュータのブラウザに関連付けられ、前記ソフトウェア・エージェントがアプレットである、請求項 23 記載のコンピュータ・プログラム製品。

【請求項 26】複数のマシンにサービスする管理サーバを有する大規模分散コンピュータ・ネットワーク内に設置され、プロセッサ及びメモリを有するコンピュータで使用されるコンピュータ読出し可能媒体内のコンピュータ・プログラム製品であって、各々が所与のマシンにおいて使用される、実行時エンジンの複数のインスタンスと、各々が 1 つ以上のタスクのセットを含む、ソフトウェア・エージェントのセットと、所与の要求に回答して、前記ネットワーク内に展開されるソフトウェア・エージェントを選択する手段であって、前記ソフトウェア・エージェントが所与のマシンの実行時エンジンにより実行され、当該マシンが、その後展開されるタスクを受信する候補であるか否かを判断する、選択手段と、を含む、コンピュータ・プログラム製品。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、大規模分散コンピュータ企業ネットワークの管理に関し、特に、こうしたネットワーク内に展開され、ローカル実行時環境で実行されるように適応化されるソフトウェア・コンポーネントを使用することにより、発見（ディスカバリ）操作を実行することに関する。「発見」は自動検出とも呼ばれる。

【0002】

【従来の技術】今日、企業は彼らのコンピュータ資源を全て企業ネットワーク上に配置したいと考えている。この目的のために、コンピュータを大規模な地理的に分散されたネットワーク環境内で接続し、こうした環境を分散形式で管理することが知られている。あるこうした管理フレームワークは、たくさんのノードを管理するサーバを含み、ノードの各々が、ローカル・ノードに特有のオブジェクト・データを記憶するローカル・オブジェクト・データベースを有する。管理される各ノードは、通常、たくさんの管理ルーチンを含む管理フレームワークを含み、これはリモート・マシンへのかなり多くの（例えば数百の）同時ネットワーク接続が可能である。管理されるノードの数が増加すると、システム保守問題も増加し、マシン故障または他の故障の可能性も増える。

【0003】通常の企業では、ノード数が増加すると問題は悪化する。これらのノードのうち、ファイル・サーバ、ネーム・サーバ、データベース・サーバなどはごく僅かであり、その多くはエンドポイント・マシンであ

る。ネットワーク・マシンの大多数は、通常の日には、ほとんど管理活動を経験しない単純なパーソナル・コンピュータ（PC）またはワークステーションである。

【0004】システム管理者は通常、システム及びネットワーク・タスクを通じて、こうした環境を管理する。これらのタスクは、管理者により特定のローカル・マシン上に構成され、ネットワーク内に分散または展開される。タスクを受信するマシンは、“展開ターゲット（deployment target）”と呼ばれる。しかしながら、ターゲット・マシンの位置及び特性は、通常、管理者により手動式に決定される。従って、例えば展開されるタスクがデータベース管理アプリケーションの場合、管理者はネットワーク内の特定のデータベース・サーバを指定しなければならない。このプロセスは煩わしく、時間がかかり、特にネットワークのサイズが増加し、数千ものマシンが接続されるようになると顕著である。もしシステム管理者が全てのターゲット・マシンを指定しない場合、システム管理タスクが不正に実現され得る。或いは、ターゲットの数及び位置が過剰に指定される場合、ネットワーク資源が不必要に消費される。

【0005】更に、こうした大規模管理環境において、ネットワーク管理者がいわゆる発見操作或いは自動検出操作の実行に関心を寄せる他の多くの理由が存在する。例えば管理者は、環境内の幾つの及びどのマシンが、ソフトウェア・プログラムの所与のバージョンを現在サポートするかを判断したいと思う。発見はまた、特定のマシンがソフトウェア・アップグレードをサポートするために、十分な資源（例えば使用可能なディスク記憶装置）を有するか否かを判断するために要求され得る。発見操作を実行する更に別の理由は、将来の企業拡大計画を容易にするために、システム・インベントリまたは資源在庫管理を実行する必要性または希望に関係し得る。従って、発見操作の性質及びタイプは多彩である。

【0006】既知の分散管理アーキテクチャでは、システム管理者は、手動操作でタスクを所与のマシン・グループに関連付ける必要なしに、配布要求を発行したり、タスクを展開したりすることはできない。同様に、こうした既知の技術は、特に数千のマシンが接続され管理されるようにネットワークが成長する環境において、システム管理及び保守を推進する広範な基本発見操作を容易にするように、簡単に適応化されなかった。

【0007】本発明は、従来技術のこれらの及び他の関連問題を解決するものである。

【0008】

【発明が解決しようとする課題】本発明の目的は、非常に多数のマシンが接続され、管理される分散コンピュータ企業環境において、発見操作を実行することである。

【0009】本発明の他の目的は、こうした発見操作のためにローカル実行時環境で実行される発見エージェントを分散コンピュータ・ネットワーク内に展開すること

である。

【0010】本発明の他の目的は、後続の操作（例えばタスク展開）を制御するために有用な所与の事実（例えばマシンまたはソース識別、特性、状態、ステータス、属性など）を発見するために、分散管理環境内に容易に展開されるソフトウェア・コンポーネントを提供することである。

【0011】本発明の他の目的は、管理者または他のユーザが手動操作によりタスクを所与のマシン・グループに関連付ける必要の無いように、ディスパッチャがタスク展開の受信候補である特定のマシンを識別するための機構を提供することである。

【0012】本発明の他の目的は、分散コンピュータ・ネットワーク内で特定のタスク展開を受信するターゲットとなる特定のマシンまたは資源を発見するための、Javaベースのソフトウェア“発見エージェント”を、分散コンピュータ・ネットワーク環境内に展開することである。

【0013】本発明の他の目的は、所与の要求に応答して、大規模分散コンピュータ・ネットワーク内に1つ以上の発見エージェントのセットを発動（ランチ）し、所与のタスクを受信するのに好適なターゲット・マシンまたは資源を識別し、突き止めることである。タスクは管理タスク、構成タスク、または任意の他のアプリケーションなどである。

【0014】本発明の他の目的は、発見の目的のためにネットワーク内にディスパッチされるソフトウェア・エージェントを、その後展開されるタスクのタイプの機能として、カスタマイズまたは調整することである。従って、ソフトウェア・エージェントは、候補マシンが展開のための潜在的ターゲットとしての資格があるか否かを、より容易に決定し得る。

【0015】本発明の他の目的は、大規模分散コンピュータ・ネットワーク内の配布ターゲットの発見を、より完全に自動化し、それによりシステム管理の費用及び複雑度を低減することである。

【0016】本発明の他の目的は、大規模コンピュータ・ネットワーク内での以後のタスク展開のために、配布ターゲットを発見するために必要となり得る最少量のコードを、最初にディスパッチすることである。

【0017】本発明の他の目的は、所与の基準を満足するワークステーションを発見するために、自己ルーティング・ソフトウェア・エージェントを分散コンピュータ・ネットワーク内に展開することである。特定の探索の間、所与のエージェントが特定のノードにおいて、自分自身をクローン化し、新たなネットワーク経路に沿って探索を続ける。

【0018】本発明の他の目的は、移動（モバイル）発見エージェントがネットワーク全体に渡りディスパッチされ、移動されるとき、大規模コンピュータ・ネットワ

ーク環境内のワークステーションに関する情報を収集することである。

【0019】

【課題を解決するための手段】本発明のこれらの及び他の目的が、大規模分散コンピュータ・ネットワーク環境における発見のために開示されるシステム、方法及びコンピュータ製品により達成される。ネットワーク環境内でサポートされる管理基盤が、好適には中央位置（例えば管理サーバ）に配置されるディスパッチ機構と、ネットワークの所与のノード上でサポートされる実行時環境とを含む。特に、実行時環境（例えばエンジン）は、好適には、分散企業環境の各管理ノード上でサポートされる分散フレームワークの一部である。

【0020】1つの好適な方法は、配布要求時に開始する。配布要求は、任意の特定タイプのシステムまたはネットワーク管理、構成または管理タスクに制限されない。要求に回答して、ディスパッチ機構は、展開のターゲットとなるマシン（すなわちターゲット・“マシン”）が、ローカル・ソース（例えば以前に収集または生成された構成情報のローカル・レポジトリ）から識別され得るか否かを判断する。こうした情報が入手可能でないか、有用でない場合、ディスパッチ機構は、展開のために好適なターゲットを突き止め、識別するようにタスクを課される、1つ以上の発見エージェントのセットをネットワーク内に展開する。これらの1つ以上のエージェントは、以後のタスク展開を容易にする情報を収集するため、ネットワーク内にファンアウトする。好適には、発見エージェントは、後に展開される特定のタスクの機能としてカスタマイズまたは調整される、小さなコードである。このカスタマイズは、探索全体を完了するために必要な時間を低減する。なぜなら、エージェントが、特定の特性の候補ノードを評価するように調整されるからである。その特性が存在しない場合、ソフトウェア・エージェントはどこか他のところに進む（或いは、自分自身をクローン化し、新たなネットワーク経路を追従する）。

【0021】特定の発見エージェントがネットワーク内のあるノードに到来すると、そのエージェントは好適には、既に存在するローカル実行時環境内にリンクされ、それによりローカル発見プロセスを開始する。発見エージェントにより実行される発見ルーチンは、ローカル・マシン（または所与の資源もしくはその上のアプリケーション）が好適なターゲットであること、ローカル・マシンが好適なターゲットでないこと、またはこの判断を下すために不十分な情報しか得られないことを発見し得る。発見プロセスの間に獲得された情報にもとづき、ソフトウェア・エージェントは、探索完了まで発見プロセスを続行するために横断されなければならない1つ以上の新たなネットワーク経路を識別する。ソフトウェア・エージェント、すなわち発見エージェントは次に、自分

自身を別のノードに発動するか、或いは必要に応じて自分自身をクローン化し、クローン化エージェントを新たなネットワーク経路を介して発動する。

【0022】候補マシンが好適なターゲットであることをソフトウェア・エージェントが発見すると、特定の識別情報（例えば確認、マシン識別子、状態識別子など）が生成される。識別情報は次に、エージェントに関連付けられるデータ記憶に保管されるか（エージェントがディスパッチ機構に戻る場合）、またはディスパッチ機構に返送される（エージェントが発見プロセスの完了時に自分自身を消滅させる場合）。こうした伝送は、単純なメッセージング技術により実行され得る。所与のネットワーク経路がすべて探索されると、発見エージェントはディスパッチ機構に戻るか、場合によっては自分自身を消滅させる。

【0023】こうして各ノードにおいて、ソフトウェア・エージェントは好適には、以前にそこに展開された実行時エンジンにより実行される。或いは、ソフトウェア・エージェントが既存のローカル資源を用いて、独立型プロセスとして実行される。ワークステーションの（ターゲット・マシンとしての）好適性が不確かな場合、ソフトウェア・エージェントは、その判断の助けとして、ディスパッチ機構または所与の他のネットワーク資源から、追加のコードを獲得し得る。こうした追加のコードは、別のソフトウェア・エージェントであり得る。

【0024】好適な発見操作は、所与のマシンまたは資源がタスク展開にとって好適なターゲットであるか否かの決定を含むが、他の発見操作も同様に実現され得る。従って、発見操作は在庫管理のために、或いはどのマシンが所与のソフトウェアのどのバージョンをサポートするかを判断するために、或いは所与のマシンまたは関連資源が所与のソフトウェアをサポートするか、または所与のタスクを実行する能力を判断するため、などを目的として実装され得る。

【0025】上述の説明は、本発明のより直接的な目的の幾つかの概要を述べたものである。これらの目的は、本発明の優れた特徴及び応用例の幾つかを表すに過ぎず、他の多くの有益な結果が本発明を多様に適用することにより、或いは本発明を変更することにより獲得され得る。従って、本発明の他の目的及びより十分な理解が、以下の本発明の好適な実施例の詳細説明を参照することにより、得られよう。

【0026】

【発明の実施の形態】図1を参照すると、本発明は好適には、数千個に上る“ノード”を含む大規模分散コンピュータ環境10内で実現される。ノードは通常、地理的に分散され、環境全体が分散的に“管理”される。好適には、管理環境（ME）は論理的に、一連の緩く接続された管理領域（MR）12に分解され、各管理領域がローカル資源を管理するための固有の管理（TME）サーバ

14を有する。ネットワークは通常、他の分散ネットワーク機能を実行する他のサーバ(図示せず)を含む。それらにはネーム・サーバ、セキュリティ・サーバ、ファイル・サーバ、スレッド・サーバ、タイム・サーバなどが含まれる。複数のサーバ14が企業内におけるアクティビティを調整し、リモート・サイト管理及び操作を可能にする。各サーバ14は多数のゲートウェイ・マシン16をサービスし、その各マシン16は複数のエンドポイント18をサポートする。サーバ14は端末ノード(TN)・マネージャ20を使用し、MR内の全てのアクティビティを調整する。

【0027】図2を参照すると、各ゲートウェイ・マシン16はシステム管理フレームワークのサーバ・コンポーネント22を実行する。サーバ・コンポーネント22は幾つかのコンポーネント、すなわちオブジェクト要求ブローカ(ORB)21、権限サービス(オーソライザ)23、オブジェクト探索サービス(ロケータ)25、及び基本オブジェクト・アダプタ(BOA)27を含むマルチスレッド実行時プロセスである。サーバ・コンポーネント22はまた、オブジェクト・ライブラリ29を含む。好適には、ORB21はオペレーティング・システムとは別に連続的に実行され、サーバ及びクライアント・プロセスの両方と、別々のスタブ及びスケルトンを通じて、プロセス間通信(IPC)機構19を介して通信する。特に、セキュア・リモート・プロシージャ呼び出し(RPC)がリモート・オブジェクトに対するオペレーションを呼び出すために使用される。ゲートウェイ・マシン16は更に、オペレーティング・システム(OS)15及びスレッド機構17を含む。

【0028】システム管理フレームワークは、各エンドポイント・マシン上でサポートされるクライアント・コンポーネント24を含む。クライアント・コンポーネント24は安価な低保守性アプリケーションの一式であり、好適には、システム管理データが持続的にそこにキャッシュまたは記憶されないという意味において、“データレス”である。この“クライアント-サーバ”形態の管理フレームワークの例は、従来技術に勝る大きな利点を有し、管理環境内へのパーソナル・コンピュータの接続性を容易にする。オブジェクト指向アプローチを使用することにより、システム管理フレームワークが、MR内の資源を管理するために要求されるシステム管理タスクの実行を容易にする。こうしたタスクは極めて多彩であり、ファイル及びデータ配布、ネットワーク使用モニタリング、ユーザ管理、プリンタまたは他の資源の構成管理などを含む。

【0029】図1に示されるような大企業では、好適には1MRにつき1つのサーバと、幾つかのゲートウェイが存在する。図3に示されるようなワークグループ・サイズの導入(例えばローカル・エリア・ネットワーク)では、単一のサーバ・クラスのマシンがサーバ及びゲ

トウェイとして使用され、クライアント・マシンは、低保守性フレームワークを実行する。このように、サーバ及びゲートウェイは単一のプラットフォームに結合され得るので、本明細書でそれらを別々に参照したとしても、本発明はそのような個別の構成に限定されない。中間サイズの導入では、MRが横に広がって成長し、追加のゲートウェイがエンドポイントの負荷を平衡化するために使用される。

【0030】サーバは、全てのゲートウェイ及びエンドポイントに勝る最上レベルの権限である。サーバは、管理領域内のあらゆるエンドポイントを追跡するエンドポイント・リストを維持する。このリストは好適には、エンドポイントを固有に識別及び管理するために必要な全ての情報を含み、それらには名前、位置及びマシン・タイプなどが含まれる。サーバはまた、エンドポイントとゲートウェイとの間のマッピングを維持し、このマッピングは好適には動的である。

【0031】上述のように、1つの管理領域当たり、1つ以上のゲートウェイが存在する。好適には、ゲートウェイはゲートウェイとして作用するように構成された完全管理のノードである。エンドポイントがログインすると、ゲートウェイはそのエンドポイントのエンドポイント・リストを作成する。ゲートウェイの役割としては、好適には、エンドポイント・ログイン要求の聴取、エンドポイント更新要求の聴取、及び(その主要タスクである)エンドポイントへのメソッド呼び出しのためにゲートウェイとして作用することが含まれる。

【0032】上述のように、エンドポイントはシステム管理フレームワーク・クライアント・コンポーネントを実行するマシンであり、ここでは低コスト・フレームワーク(LCF)と呼ぶことにする。LCFは図7に示される2つの主要部品、すなわちLCFデーモン24A、及びアプリケーション実行時ライブラリ24Bを有する。LCFデーモン24Aの役割は、エンドポイント・ログインと、アプリケーション・エンドポイント実行可能コードの作成である。一旦実行可能コードが作成されると、LCFデーモン24Aはもはやそれと対話しない。各実行可能コードは、ゲートウェイとの以後の全ての通信を処理するアプリケーション実行時ライブラリ24Bとリンクされる。

【0033】好適には、サーバ及び各ゲートウェイはコンピュータまたは“マシン”である。例えば、各コンピュータは、AIXオペレーティング・システム(好適にはバージョン3.2.5以上)を実行するRISC System/6000(商標)である。好適な代替マシンには、Novell UnixWare 2.0を実行するIBM互換PC(x86以上)、AT&T UNIX SVR4 MP-RAS Release 2.02以上を実行するAT&T 3000シリーズ、DG/UXバージョン5.4R3.00以上を実行するData G

eneral AViiON シリーズ、HP/UX 9.0
0乃至HP/UX 9.05を実行するHP 9000/7
00及び800シリーズ、SVR4バージョンR40V
4.2を実行するMotorola 88Kシリーズ、
Solaris 2.3または2.4を実行するSun
SPARCシリーズ、またはSunOS 4.1.2また
は4.1.3を実行するSun SPARCシリーズが
含まれる。勿論、他のマシン及びオペレーティング・シ
ステムも同様に、ゲートウェイ及びサーバ・マシンとし
て使用され得る。

【0034】各エンドポイントもまたコンピュータであ
る。本発明の好適な1実施例では、ほとんどのエンドポ
イントはパーソナル・コンピュータ（例えばデスクトッ
プ・マシンまたはラップトップ）である。このアーキテ
クチャでは、エンドポイントは強力なまたは複雑なマシ
ン或いはワークステーションである必要はない。1つま
たは複数のエンドポイントが、例えばIBM ThinkPad（商標）マシン、または他のIntel x8
6またはPentium（商標）ベースのコンピュータ
などの、Windows 95またはそれ以降のオペレ
ーティング・システムを実行するノートブック・コンピュ
ータであり得る。OS/2（商標）オペレーティング・
システムのもとで動作するIBM（商標）マシンまたは
IBM互換マシンも、エンドポイントとして使用され得
る。エンドポイント・コンピュータは好適には、Netsca
pe NavigatorまたはMicrosoft Internet Explorerなど
のブラウザを含み、インターネット、イントラネットま
たは他のコンピュータ・ネットワークを介してゲートウ
ェイに接続され得る。

【0035】本発明の好適な好適な実施例は、図4に示
されるような企業環境において実現される。後述するよ
うに、ソフトウェア“発見エージェント”のセットが、ネ
ットワーク内の中央位置（例えばマネージャ14）にお
いて、または複数の位置（例えばゲートウェイ16）に
おいて使用可能であり、そこでは構成タスクなどの管理
タスクが指定、構成または展開される。ソフトウェア・
エージェントは、（後述のように）ディスパッチ機構か
らディスパッチされ、ネットワーク環境全体を通じて移
動すると言う意味において「モバイル」である。

【0036】一般に、モバイル・ソフトウェア・エー
ジェントはネットワークを横断し、いわゆる発見操作を実
行する。発見操作のタイプは極めて多彩である。例えば
特定の発見操作は、一旦発見アプリケーションが開始さ
れると、ユーザにより管理資源において、通常のグラフ
ィカル・ユーザ・インタフェース（GUI）を介して開
始され得る。この操作は単に、（所与のエージェントが
実行される）マシンのセットの各々に問い合わせ、マシ
ン・タイプを決定するための、1つ以上の発見エー
ジェントを発行するだけであつたりする。発見操作は、所
与のマシンに関連付けられる資源のリストを識別し得る。

発見操作は、所与のマシンが特定タイプの資源を有する
か否かを識別し得る。別の発見操作は、マシンまたは特
定の関連資源が所与の特性を有するか否かを発見するた
めに、単にマシンに問い合わせただけであつたりする。後
者の状況の例は、特定の資源（例えばディスク・ドライ
ブ・パーティション）が、特定の定義基準（例えば記憶
空間）に合致するか否かを判断する発見操作が、所与の
マシンにおいて開始される場合に相当する。このよう
に、発見操作は極めて一般的であつたり、非常に特定の
であつたりし、所与の操作が既存の状態（例えば既存の
資源またはそれらの動作状態）に関連したり、或いはマ
シンが将来（例えば計画されたシステム拡張のため
に、）他の資源をサポートできるか否かを判断したりす
る。後者の状況の例は、ネットワーク管理者が既存のマ
シンのインベントリを実行し、ソフトウェア・アップグ
レードを要求するマシンを判断する場合に相当する。

【0037】例えば、ネットワーク管理者が分散コンピ
ュータ環境内の所与の資源をモニタしたい場合、勿論、
発見エージェントの性質が直接、それが作用するモニタ
リング・コンポーネントにリンクされる。かくして、特
定のオペレーティング・システムからだけ使用可能な特
定のメトリックをモニタしたいモニタリング・コンポー
ネントに対しては、発見エージェントはこうしたシステム
上でだけ発見の成功を報告する。

【0038】一部のモニタリング・コンポーネントは、
資源の幾つかのインスタンスが任意の所与のコンピュ
ータ上に存在し得るような特定の資源タイプをモニタし
ようとし得る。こうした資源には、例えばディスク・ドラ
イブまたはそのコンポーネント（すなわちファイル・シ
ステム）、特定タイプのプロセス、及び特定タイプのロ
グ・ファイルなどが含まれる。これらのコンポーネント
のための発見エージェントは、所与のコンピュータ上の
こうした資源のインスタンスを見い出すように、また要
求に応じて、見い出された各資源に対するモニタリング
・コンポーネントのコピーの実体化（インスタンス化）
を生じるように設計され得る。

【0039】一部の発見エージェントは、絶えずシステ
ムの状態をモニタするように設計され、それにより動的
に出現し、消滅する資源が追跡され得る。こうしたエー
ジェントは通常、特定の単純なスケジューリング・メト
リックに従う資源インスタンスを求めて走査する。こう
した資源の例は、クライアントと特定のソフトウェア・
サーバ・プロセス間のアクティブ接続である。

【0040】発見プロセスの結果、しばしば発見された
情報が収集され、ディスパッチャまたは他の位置に返却
されることが望まれる。ユーザに返却される特定の情報
は、開始された発見操作のタイプに依存する。こうした
情報の提示及びフォーマットは、単なる設計の問題であ
り、本発明を制限するものではない。例えば、発見操
作は単に、所与のソフトウェア・ルーチンの“バー

ジョン x、y を有する全てのマシンの識別を探索し、返却される情報は単に、スクロール式リストボックスまたは他の既知の GUI 構造内に表示される、識別された資源の順序づけされたリストであつたりする。

【0041】後述の実施例では、発見操作が特定のタスク展開を受信する管理環境内のマシンを決定する。タスク展開は例えば、分散ネットワーク全体に渡り、資源を管理するために使用される分散モニタリング・アプリケーションである。この例について詳述するが、当業者であれば、特定の発見操作の性質、タイプ及び特性が、状況に応じて、極めて多様であることが理解できよう。本発明はいかなる特定の発見操作、またはこうした操作のいかなる定義済みセットにも限定されるものではない。

【0042】この実施例では、環境内で展開される特定のタスクが指定され得るが、ターゲット・マシンは容易に突き止められないかも知れない。こうしたケースでは、この情報を決定するために、適切な“発見”エージェントが識別され、ディスパッチされる。発見エージェントが初期調査位置においてターゲット・マシンを見い出さない場合、発見エージェント（またはそのクローン）はネットワークを移動し、探索を続ける。発見エージェントは好適には、ディスパッチ位置において受信される情報にもとづき、また任意的に各調査位置から収集される情報から、ネットワークを通じるその経路を選択する。特定の“経路”は通常、ソフトウェア発見エージェントがネットワークを通じて移動するとき変化する。なぜなら、特定のノードから収集される情報は、発見エージェントを所与の態様で宛先変更（リダイレクト）するからである。

【0043】説明の都合上、こうした発見方法が図4に示される大規模分散企業環境において実現されるものとするが、これは本発明を制限するものではない。この例では、マネージャ14がそれに関連付けられるソフトウェア・エージェント37のセットを有するディスパッチ機構35を含む。或いは、ディスパッチ機構35が構成可能なソフトウェア・タスク39のセットを含み、そこから1つ以上のエージェントが構成される。マネージャ14は好適にはデータベース43を含み、これが分散コンピュータ環境内で管理されるように設計される全てのマシンのリストを識別する情報を含む。ディスパッチ機構自体が複数のノードに渡り分散されてもよい。

【0044】ゲートウェイ・ノード16の少なくとも幾つかと、端末ノード18の少なくとも幾つか（またはその特定の定義済みサブセット）が、配布サービスを介して特定のノードにダウンロードされた実行時エンジン41を含む。実行時エンジン41はソフトウェア・エージェントのための実行時環境を提供する。特定の配布技術が、米国特許第5838918号明細書で述べられるような加入プロセスを含み得るが、これに限られるものではない。或いは診断エンジンが、1998年6月3日付

けの米国特許出願第89964号で述べられる技術により様々なノードに配布され得る。

【0045】上述のように、本発明は自動的に1つまたは複数のソフトウェア発見エージェントを展開して、所与の発見操作（例えば特定のマシン、その上の資源などを突き止める）を実行し、管理者または他のシステム・エンティティにより指定される管理タスク（または他の何らかのサービス）を容易にする。好適には、ソフトウェア・エージェントは、エージェントが到来するノードに配置される実行時エンジンにより実行されるソフトウェア・コンポーネント（コード）である。或いは、ソフトウェア・エージェントはローカル資源を使用する独立型のアプリケーションとして実行される。更に別の態様では、ソフトウェア・エージェントは、ホスト・プラットフォームを調査して発見操作を実行するエンジンを制御する（この例では、ターゲット・タスク展開を受信するホストの好適性を決定する）。

【0046】典型的な実施例では、実行時エンジンとソフトウェア・エージェントの両方がJavaで作成される。周知のように、Javaはソフトウェア・プログラムを開発、テスト及び保守するために使用される、オブジェクト指向マルチスレッド型の移植性のあるプラットフォーム独立のセキュア・プログラミング環境である。Javaプログラムは、インターネットのマルチメディア情報検索システムであるワールド・ワイド・ウェブ上で広く使用されている。これらのプログラムは、フル機能の対話式独立型アプリケーション、及びJavaが使用可能なウェブ・ブラウザにおいて実行されるアプレットとして知られる小さなプログラムを含む。

【0047】本発明の特定の実施例では、ソフトウェア・エージェントはJavaアプレット（例えばJava“クラス”・ファイルのセットから成る）であり、実行時環境はウェブ・ブラウザに関連付けられるJava仮想マシン（JVM）を含む。この実施例では、ネットワークの様々なノードはインターネット、イントラネット、または他のコンピュータ・ネットワークの一部である。

【0048】管理者が展開用のタスクを構成するとき、ディスパッチ機構が（好適にはそのタスクまたはその特定の特性にもとづき、）適切なJavaクラス・ファイルをコンパイルし、アプレットを（ソフトウェア・エージェントとして）ネットワーク内にディスパッチする。ネットワークのサイズ、構成及びトポロジに応じて複数のエージェントがディスパッチされ得る。次に、各アプレットが候補ノードに配置されるJVM上で実行され、そのノードがタスクの展開のために適切なターゲットか否かを判断する。

【0049】図5は、本発明に従う発見ルーチンの例である。このルーチンの一部はしばしば、異なる時刻に、また異なる制御状況の下で発生し得る。それらは単に説

明のために一緒に示されているだけである。

【0050】ルーチンはステップ30で開始し、実行時エンジンが様々なノードに配布される。この実施例では、実行時エンジンは上述のように、LCF実行時ライブラリ24Bの一部であり得るが、通常は、実行時エンジンは前のノード構成タスクの間に、実行時ライブラリより前に展開される。いずれの場合にも、これらの実行時エンジンは集合的に、企業環境の管理基盤の一部を形成する。一旦管理基盤が用意されると、実際の操作ルーチンが開始する。

【0051】ステップ32では、ディスパッチ機構35において、例えばタスク展開要求、メソッド呼び出しなどの所与のオカレンスが生成されたか、或いはネットワーク内のどこか別の場所から受信されたか否かを判断するテストが実行される。上述のタスク展開の場合、管理者が通常のGUIを用いて所望のタスク構成を実行する。タスクを構成または指定する技術自体は、本発明の一部ではない。ステップ32は勿論、任意の所与の機能を表し得、単にタスク展開だけに限られるものではない。しかしながら、説明を容易にするために、発見プロセスがこうした展開に付随すると仮定することにする。

【0052】ステップ32のテスト結果が否定の場合、ルーチンは図示のように巡回する。しかしながら、ステップ32のテスト結果が、展開されるタスクが指定されたことを示すと、この方法はステップ34に移行する。

【0053】ステップ34で、発見可能か否か、すなわち発見プロセスがシステムまたはシステム管理者により、指定の性質と一緒にイネーブルされたか否かのテストが実行される。制限的な意味ではないが、管理者はGUI、コマンド・ライン・インタフェース(CLI)または任意の既知のインタフェース技術を用いて、発見プロセスをイネーブルする(更に、特に発見操作を指定する)。ステップ34のテスト結果が否定の場合、ルーチンは終了する。ステップ34で発見プロセスがイネーブルされた場合、ルーチンはステップ36に移行し、レポジトリ43(例えば管理サーバ内)に問い合わせることにより、ターゲット・マシン及びそれらの特性(例えば位置、状態、ステータス、構成など)が、既に発見または指定されたか否かを判断する。ステップ36のテスト結果が肯定の場合、ルーチンはステップ38で情報をディスパッチ機構に返却し、ステップ40で必要に応じて返却されたデータが実体化(インスタンス化)される。このパスが実行される場合、発見が必要とされないため、ルーチンは終了する。

【0054】しかしながら、ステップ36の問い合わせの結果が、必要な情報がレポジトリから入手可能でないことを示すと(例えばこうしたレポジトリが存在しなかったり、配布を調整するために必要な特定の情報が収集されなかったり、情報が所与のエンジニング・ファクタにより陳腐化したりすることによる)、ルーチンはステッ

プ42に移行する。そこでは、展開されるタスクが解析され、1つまたは複数の探索特性が識別される。このステップは自動的に実行されるか、或いはユーザが(例えばGUIを介して)指定する情報により制御され得る。例えば展開されるタスクが、データベース・サーバ上でサポートされるデータベース管理タスクの場合、ステップ42は探索プロセスを容易にするために、候補マシンの所与の特性を識別する。この例では、その特性は「常駐データベース・サーバ・ソフトウェアを有するマシン」などである。このプロセスを容易にするため、GUIはマシン、資源またはそれらの属性との関連づけを形成するために選択されるアイコンその他のビジュアル・オブジェクトを表示し得る。勿論、指定または選択のために任意の便利な機構を使用し得る。展開されるタスクの1つ以上の特性を識別することにより、本機構は候補マシン上でより効率的に特定の指定ハードウェア、ソフトウェアまたは他のコンポーネントを探し当てるように、ソフトウェア・エージェントを調整またはカスタマイズし得る。

【0055】この目的のために、ルーチンはステップ44に続き、ステップ42で導出された所与の特性、他のユーザ選択基準もしくはシステム選択基準、または履歴データなどの他の情報にもとづき、適切なソフトウェア・エージェントを選択、構築またはサブクラス分類する。ここではステップ44の選択プロセスは、1つ以上のソフトウェア・タスクを、この目的のための「カスタム」ソフトウェア・エージェントに編成するステップを含み得る。このように、本発明は既存のソフトウェア・エージェントの他に、「オンザフライ式」に作成または生成されるエージェントの使用を含む。

【0056】ステップ46で、ソフトウェア・エージェントがネットワーク内に展開される。このソフトウェア・エージェントは、コードが局所の実行状況に置かれ、特定の発見操作を実行するように制御され得るように設計された適切なルーチンを含む。ステップ46はネットワークのトポロジに応じて、(同時にまたは継続的にディスパッチされる)複数のエージェントの展開を含み得る。上述のように、各ソフトウェア・エージェントはモバイル発見エージェントであり、その目的は配布情報を発見することである。これにより発見エージェント展開ルーチンが完了する。

【0057】図6のフローチャートは、特定ノードにおける発見操作を示す。ルーチンは、所与のソフトウェア・エージェントがノードに到来するとき、ステップ50で開始する。勿論、複数のエージェントがネットワーク内にディスパッチされ得るので、図6に示されるルーチンは、ネットワーク内の多くの異なるノード上で同時に実行され得る。ステップ52で、発見エージェントがローカル自動化エンジンにリンクされる。こうしたリンクは通常、ソフトウェア・エージェントを実行時環境に結

合するステップを含む。次にステップ 54 で、ローカル発見プロセスが開始される。ステップ 56 では、テスト下のシステムが所与の基準に適合するか否かのテストが実行される（この基準は好適には、上述のカスタマイズ・プロセスを通じて指定される）。ステップ 56 のテスト結果が不確定の場合、ルーチンは図示のように巡回する。しかしながら、ステップ 56 のテスト結果が、マシン（またはその所与のコンポーネント）がソフトウェア・エージェントにより指定された探索基準を満足することを示す場合、ルーチンはステップ 58 に分岐する。

【0058】ルーチン内のこの時点において、ソフトウェア・エージェントは、ディスパッチ機構により要求または所望される適切な情報を収集、編成または生成する。この情報には、例えば、ホスト・プラットフォームを識別する情報またはデータ、その位置及び他の識別特性、動作コンポーネントの現状態を識別する情報などが含まれる。情報のタイプは、展開されるタスクに応じて、或いはより一般的には、発見操作自体の性質に応じて変化し得る。ステップ 60 で、所与のネットワーク経路に渡る探索の完了時に、ソフトウェア・エージェントが自己消滅型か否か、すなわちソフトウェア・エージェントが自分自身を消滅または“終了 (kill)” するか否かのテストが実行される。ステップ 60 のテスト結果が、ソフトウェア・エージェントが自己消滅型であることを示す場合、ルーチンはステップ 62 に進み、例えばローカル・メッセージ機構を使用して、識別情報をネットワークを介して返送する。ソフトウェア・エージェントが自己消滅型でない場合、ステップ 64 で、識別情報がそのソフトウェア・エージェントに関連付けられるデータ記憶に書込まれる。こうした情報は、後にソフトウェア・エージェントがディスパッチ機構に戻るとき、吐き出される。典型的なデータ記憶は、エージェント・コード自身内に配置される。或いは、識別情報がプラットフォームに記憶され得る。

【0059】制御は次にステップ 66 に移行する。このステップは、ステップ 56 の結果がプラットフォームが探索基準に適合しないことを示す場合にも、メイン処理ループ内で実行される。ステップ 66 で、他の追加のネットワーク経路が横断されるべきか否かのテストが実行される。この判断基準は、例えば特定のマシンがエンド

ポイントであるか否かといった具合に簡単であったり（エンドポイントの場合、横断は要求されない）、或いはより複雑であったりする。テスト結果が否定の場合、ステップ 68 で、ルーチンはソフトウェア・エージェントを消滅するか（ステップ 60 のテスト結果が肯定の場合）、或いはソフトウェア・エージェントがディスパッチ機構に向けて返送される。しかしながら、ステップ 66 のテスト結果が肯定の場合、ソフトウェア・エージェントにより横断される追加のネットワーク経路が存在する。

【0060】ルーチンは次にステップ 70 に移行し、ソフトウェア・エージェントが探索を続けるために、クローン化されるか否かをテストする。例えばソフトウェア・エージェントが他の診断目的のために有用な場合などの特定の状況下では、ローカル発見が完了された後に、ソフトウェア・エージェントをプラットフォームに保持することが望ましい。それにより、例えばノードにおける将来の発見操作が、既に存在する実行済みのエージェント（またはその一部）を有することにより、単純化され得る。

【0061】このように、ソフトウェア・エージェントは一般に、ディスパッチャに戻る能力、ノードに留まる能力、或いは自分自身の別のインスタンスをクローン化して発動し、探索を続行する能力を含む。ステップ 70 のテスト結果が肯定の場合、ソフトウェア・エージェントはステップ 72 でクローン化され、次にステップ 74 で、識別された経路を介して発動される。このルーチンは、全てのソフトウェア・エージェントがそれら自身を消滅するか、或いはディスパッチャに戻るまで、繰り返し実行される。ディスパッチャに返送された情報は、好適には将来の探索における参照目的のために（例えば探索分野を狭めるために）記憶される。

【0062】本発明は、環境内で展開される特定のタスクのために、ターゲット・ワークステーションの位置及び特性を発見する機構を提供する。タスク自身はソフトウェア・エージェント、または任意の他のタイプのアプリケーション、プロセスまたは他のルーチンである。先行する発見があった場合、そこから導出された情報が展開を容易にするために使用され得る。しかしながら、こうした情報が入手可能でないか、（例えば陳腐化のために）有用でない場合、最初に 1 つ以上の発見エージェントが要求情報を発見するために発動される。次に、発見された情報がディスパッチャに返却され、タスクのターゲット先への配布を容易にするために使用される。

【0063】当業者であれば、ディスパッチャが探索方法を 1 つまたは複数の態様により指令し、要求されるソフトウェア・エージェントの数、またはターゲット・ノードのリストを生成するために訪問されなければならないノードの数を低減することが理解できよう。次に、ソフトウェア・エージェントが汎用“ターゲット”領域に展開され、そこから特定のターゲット位置が識別される。

【0064】好適な 1 実施例では、ソフトウェア・エージェントは、大規模分散コンピュータ・ネットワーク内の適切なシステムに経路指定可能なタスクのセットから成るオブジェクトである。故障を診断または修正するために必要であれば、タスクのセットを一緒に結合することができる。各ノードにおいて、ソフトウェア・エージェントは好適にはその内部に組み込まれるか、以前に展開された実行時環境により実行される。従って、発見機能の大部分（すなわち実行時エンジン）が既に評価され

るシステムに存在するので、ネットワーク・トラフィックが更に最小化される。

【0065】一旦ターゲット・マシンが識別されると、タスクがこれらのマシンに展開されるか、他の所与のアクションが実行される。例えば、ターゲット・マシンの“マップ”がディスパッチャに、または他の別の場所に記憶され、以後の続く展開を容易にする。従って、発見機構はリサーチまたは他の目的のために、ネットワーク環境のトポロジを“チャート化”または“マップ”するために有用である。タスクが後に展開されるとき、被管理ネットワークにおいてタスクの受信を要求される領域にだけ、展開がフォーカスされる。このことは帯域幅を大幅に低減し、それによりネットワーク資源を節約する。

【0066】エージェントによっては、ノードが好適なターゲットであるか否かを判断するために必要なコードを持っていないことがある。エージェントは、ローカル発見プロセスを実行するために必要なコードを有するか、またはそのための追加のコードの要求をディスパッチ機構に送信し得る。追加のコードは他のソフトウェア・エージェントであり得る。

【0067】ソフトウェア・エージェントは好適には、ターゲット・マシンを発見するために、またはローカル発見プロセスに関連付けられる特定のタスクを実行するために必要な、最小量のソフトウェア・コードである。発見機能の一部をエンジン内に分配することにより、ネットワーク帯域幅が節約される。なぜなら、少量のコードだけがターゲット・サイトにディスパッチされる必要があるからである。このことが更に、大企業環境内のシステム管理の複雑性及びコストを低減する。

【0068】好適には、各エンドポイント上で実行されるクライアント・クラス・フレームワークは、管理タスクを実行する準備が整った、しかし（通常アイドル状態であるので）ほとんどマシン資源を消費しない低保守性、低コストのフレームワークである。各エンドポイントは好適には、特定のシステム管理タスクが実現または実行される前後に、システム管理データがそこに記憶されないといった意味において、“データレス”である。このアーキテクチャは、数十個のサーバ、数百個のゲートウェイ、及び数千個のエンドポイントを有する企業の合理的区分化を有利に可能にする。各サーバは通常、200個までのゲートウェイをサービスし、各ゲートウェイは数千個のエンドポイントをサービスする。フレームワーク・レベルでは、エンドポイントへのまたはエンドポイントからの全ての操作が、ゲートウェイ・マシンを通過する。多くの操作において、ゲートウェイは透過的であり、それは要求を受信し、ターゲットを決定し、要求を再送し、結果を待機し、結果を呼び出し元に返送する。各ゲートウェイは複数の同時要求を処理し、任意の数のゲートウェイが企業内に存在し得、その正確な数は使用可能な資源や、サービスされる必要があるエンドポ

イントの数などの、多くの要因に依存する。

【0069】好適な実施例では、これらの及び他の目的が、1つ以上の管理領域内に編成されるコンピュータ資源を含む大規模分散企業内で達成され、各領域が1つ以上のゲートウェイ・マシンをサービスする管理サーバにより管理される。そして各ゲートウェイ・マシンが、複数のエンドポイント・マシンをサービスする。上述のように、システム管理フレームワークが好適にはゲートウェイ・マシン及び1つ以上のエンドポイント・マシン上に“分散”され、システム管理タスクを実行する。上述の環境が好ましい訳であるが、当業者であれば、本発明の概念がより小規模な分散クライアントサーバ・ネットワーク環境においても実現され得ることが理解できよう。従って、本発明は特定の実施例において述べられたような、特定の大規模分散コンピュータ環境に限られるものではない。

【0070】本発明の好適な1実施例は、コンピュータのランダム・アクセス・メモリ内に存在するコード・モジュール内の命令のセットとして実現される。コンピュータにより要求されるまで、命令のセットは、例えばハード・ディスク・ドライブ内や、光ディスク（CD-ROMにおいて使用される）やフロッピー・ディスク（フロッピー・ディスク・ドライブにおいて使用される）などの、取り外し可能メモリ内などの、別のコンピュータ・メモリ内に記憶されるか、或いはインターネットを介してダウンロードされ得る。

【0071】更に上述の様々な方法が、ソフトウェアにより選択的に活動化または再構成される汎用コンピュータ内で実現され得るが、当業者であれば、こうした方法がハードウェア、ファームウェア、または要求方法を達成するように構成されたより特殊な装置により実現され得ることが理解できよう。

【0072】更に、本発明は特定のネットワーク環境内の好適な実施例に関して述べられてきたが、当業者であれば、本発明がその趣旨及び範囲内において、変更により、他の異なるネットワーク・アーキテクチャにおいても実現され得ることが理解できよう。更に、本発明の診断技術は、任意の分散ネットワーク環境において有用であろう。

【図面の簡単な説明】

【図1】本発明が実現される大規模分散コンピュータ企業環境を単純化して示す図である。

【図2】フレームワーク機能がゲートウェイ及びそのエンドポイントを介して、管理領域内で分散される様子を示す、好適なシステム管理フレームワークのブロック図である。

【図3】サーバ及びゲートウェイ機能が同一マシン上でサポートされる、企業の小規模なワークグループの例を示す図である。

【図4】本発明の好適な方法を実行するために使用され

21

る管理基盤を有する、分散コンピュータ・ネットワーク環境を示す図である。

【図5】コンピュータ・ネットワーク内の配布要求に応答して、ソフトウェア発見エージェントを展開する好適な方法を示すフロー図である。

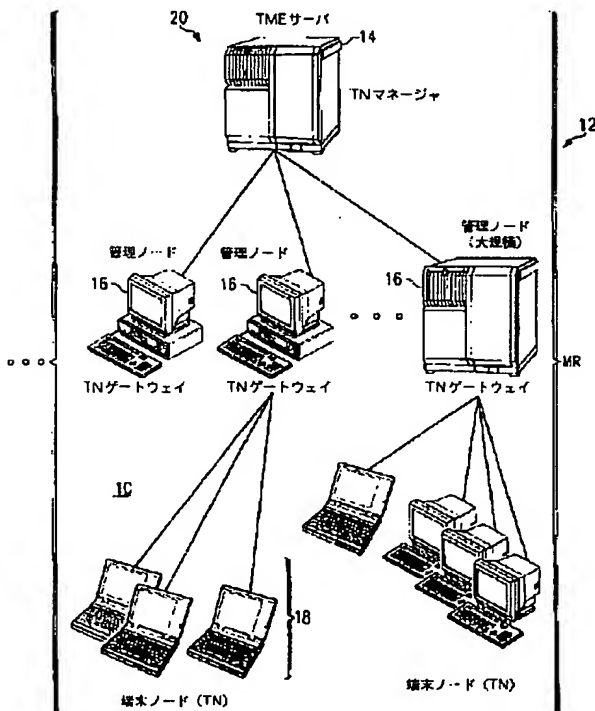
【図6】本発明の好適な実施例に従うソフトウェア・エージェント・ローカル発見機構のフロー図である。

【図7】システム管理フレームワークのLCFクライアント・コンポーネントのブロック図である。

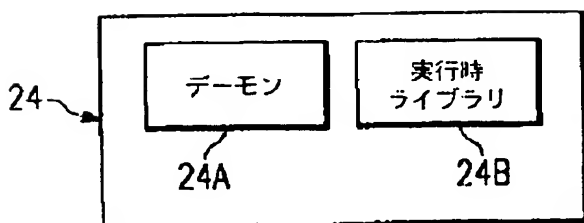
【符号の説明】

- 10 大規模分散コンピュータ環境
- 12 管理領域 (MR)
- 14 管理 (TME) サーバ
- 15 オペレーティング・システム

【図1】



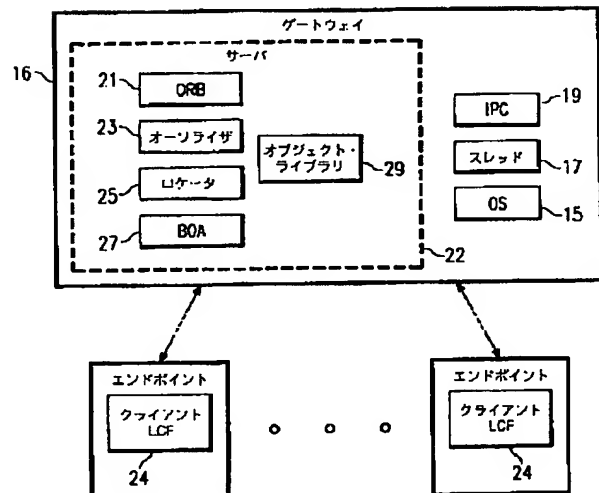
【図7】



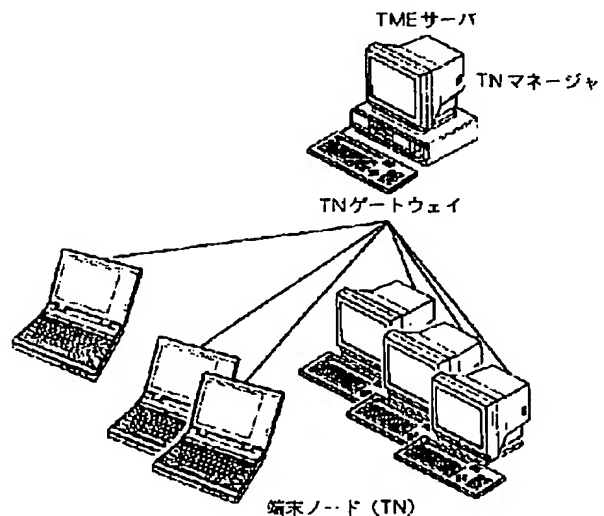
22

- 16 ゲートウェイ・マシン
- 17 スレッド機構
- 18 エンドポイント
- 19 プロセス間通信 (IPC) 機構
- 20 端末ノード (TN) ・マネージャ
- 21 オブジェクト要求ブローカ (ORB)
- 22 サーバ・コンポーネント
- 23 権限サービス (オーソライザ)
- 25 オブジェクト探索サービス (ロケータ)
- 27 基本オブジェクト・アダプタ (BOA)
- 29 オブジェクト・ライブラリ
- 35 ディスパッチ機構
- 37 ソフトウェア・エージェント

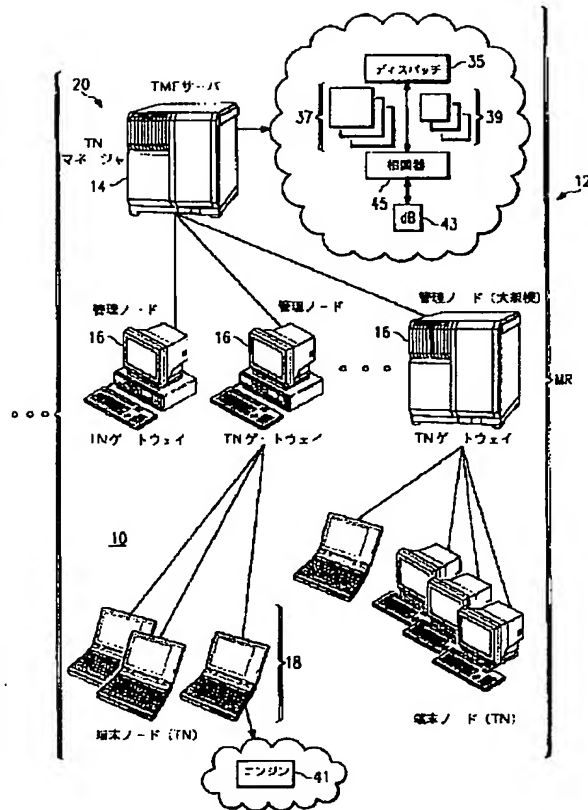
【図2】



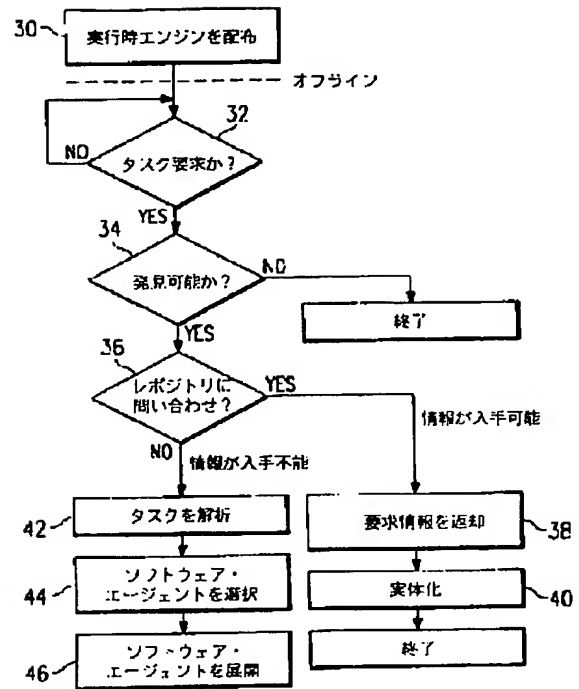
【図3】



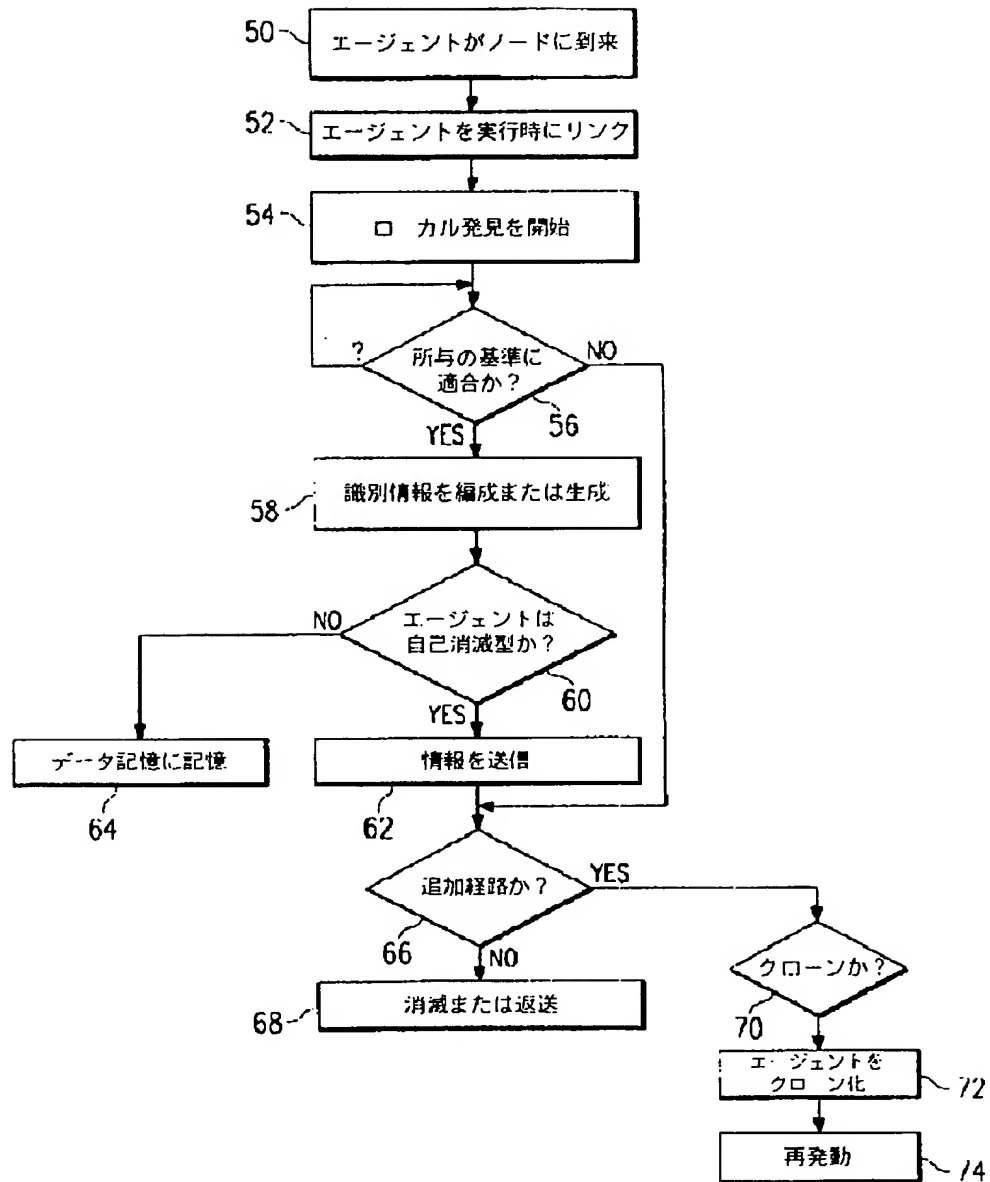
【図 4】



【図 5】



【図6】



フロントページの続き

(72) 発明者 ブライアン・ジェイ・ベター
 アメリカ合衆国78717、テキサス州オース
 ティン、クレセント・ハイツ・トレイル
 17217

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ **BLACK BORDERS**
- ☐ **IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- ☐ **FADED TEXT OR DRAWING**
- ☐ **BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- ☐ **SKEWED/SLANTED IMAGES**
- ☐ **COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- ☐ **GRAY SCALE DOCUMENTS**
- ☐ **LINES OR MARKS ON ORIGINAL DOCUMENT**
- ☐ **REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- ☐ **OTHER:** _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.